On Unconditionally Secure Computation with Vanishing Communication Cost[1]

Ye Wang*, Shantanu Rane†, Wei Sun† and Prakash Ishwar*

***Abstract*— We propose a novel distortion-theoretic approach to a secure three-party computation problem. Alice and Bob have deterministic sequences, and Charlie wishes to compute a normalized sum-type function of those sequences. We construct three-party protocols that allow Charlie to compute the function with arbitrarily high accuracy, while maintaining unconditional privacy for Alice and Bob and achieving vanishing communication cost. This work leverages a striking dimensionality reduction that allows a high accuracy estimate to be produced from only a random subsampling of the sequences. The worst-case distortion of the estimate, across all arbitrary deterministic sequences of any length, is independent of the dimensionality (length) of the sequences and proportional to inverse square root of the number of samples that the estimate is based upon.**

I. Introduction

We consider a secure three-party computation problem, where Alice and Bob have deterministic sequences x^n and y^n respectively, and Charlie wishes to compute a normalized sum-type function of the form $f_n(x^n, y^n) := (1/n)\sum_{i=1}^n f_1(x_i, y_i)$. The objective is to construct a three-party protocol that securely computes the function with high accuracy and low communication cost. We assume that the parties are semi-honest (passive), which means that they will correctly follow the steps of the protocol, but will attempt to infer the maximum possible information about each other's sequences from the data available to them. We require unconditional privacy, which means (in a strong statistical sense) that Alice and Bob are unable to infer any information about each other's sequences and that Charlie is unable to infer any information about both sequences (x^n, y^n) other than what can be inferred from his function estimate $\hat{F}_n(x^n, y^n)$. Figure 1 roughly illustrates our problem setup.

Unlike many other secure multi-party computation formulations (such as [1], [2], [3], [4]), which aim to make the probability of error, $\Pr\left[\hat{F}_n(x^n, y^n) \neq f_n(x^n, y^n)\right]$, equal to zero or negligible, we consider a novel distortion-theoretic approach that aims to minimize the maximal expected absolute error

$$\max_{x^n, y^n} E\left[\left|\hat{F}_n(x^n, y^n) - f_n(x^n, y^n)\right|\right],$$

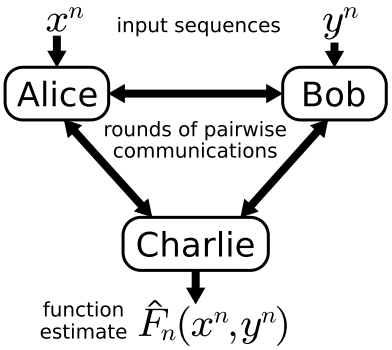

Fig. 1. Alice and Bob are given deterministic sequences x^n and y^n. The three parties then execute a protocol consisting of multiple rounds of local computations and pairwise communications. At the end of the protocol, Charlie produces an estimate of a function of the sequences, $\hat{F}_n(x^n, y^n)$.

where the expectation is with respect to any randomness inherent to the protocol in generating the estimate $\hat{F}_n(x^n, y^n)$. The distortion is the worst-case expected absolute error across the deterministic sequences (x^n, y^n). The communication cost is given by the number of bits of transmission required by a protocol divided by the length of the sequences n. Our main result is the construction of unconditionally private protocols that allow Charlie to estimate any normalized sum-type function $f_n(x^n, y^n)$ with both vanishing distortion and vanishing communication cost as $n \to \infty$. While the (perfectly) secure multi-party computation techniques of [1] can be used to compute any normalized sum-type function without error (zero distortion), they require $O(n)$ transmissions (see Section III-E) and hence have non-vanishing communication cost.

The key to our result is the realization that any normalized sum-type function can be evaluated accurately even after drastically reducing the dimensionality of its inputs (x^n, y^n). Although there are elegant dimensionality reduction results which show that distances can be approximately preserved by mapping high-dimensional signals into a low-dimensional subspace [5], [6], they are within a *centralized* computation context and without privacy constraints. We consider a much more basic dimensionality reduction that is achieved by a simple random subsampling. It was shown in [7] that an accurate estimate of the joint type of (x^n, y^n) can be produced from a randomized subsampling of the sequences (x^n, y^n). We produce a simpler alternative analysis of the work in [7], which allows us to analyze the expected distortion, and apply this result to create accurate estimates of normalized sum-type functions in a manner that is both

[1]This material is based upon work supported in part by the US National Science Foundation (NSF) under awards (CAREER) CCF–0546598 and CCF–0915389. Some of this work was conducted when Ye Wang was a summer intern at MERL. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. *Department of Electrical and Computer Engineering at Boston University, Boston, MA 02215, Emails: {yw,pi}@bu.edu. †Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA 02139, Emails: {rane,weisun}@merl.com.

secure and communication efficient. The randomization in the subsampling is crucial for overcoming the worst-case distortion criterion, and thereby achieving vanishing distortion. Subsampling by a factor much smaller than n allows us to use fewer invocations of the secure computation primitives of [1] while securely producing the function estimate with vanishing communication cost.

It is important to highlight the distinction between a vanishing error-probability criterion and a vanishing expected-distortion criterion. Distributed computation of the joint type with a vanishing error-probability needs a strictly positive communication bitrate which does not vanish with increasing blocklength [7], [8], [9], whereas the bitrate vanishes with blocklength for vanishing expected-distortion [7]. Our distortion-theoretic approach to secure multi-party computation thus trades exact computation for arbitrarily high accuracy in order to gain the advantage of vanishing communication cost. This makes our work particularly relevant to applications where data size is overwhelming and only a highly accurate, but not exact, computation is necessary.

A couple of examples of potential applications are secure computation of statistics in distributed databases and distributed biometric authentication. In the first example, the sequences of Alice and Bob are viewed as a distributed database from which Charlie wishes to extract a joint statistic, represented by the normalized sum-type function, without requiring Alice or Bob to reveal any additional information about their data. Our approach allows Charlie to securely compute the statistic with arbitrarily high accuracy while attaining vanishing communication cost.

In the biometric authentication problem, Charlie wishes to verify that Alice's biometric sequence x^n is "close" to a reference biometric sequence y^n held by the authentication authority Bob, without requiring Alice or Bob to directly reveal their sequences to any other party. By having Charlie compute a normalized sum-type function, where f_1 is an appropriately chosen metric, Charlie can accurately compute the "closeness" of Alice's biometric to the reference held by Bob in order to decide whether to accept or reject the identity assertion by Alice. Recent work [10] has taken a cryptographically secure approach in computing Hamming distance and sum of squared errors for fingerprint feature vectors. Our approach can compute a much broader set of metrics with arbitrarily high accuracy, unconditional privacy and vanishing communication cost.

II. Problem Formulation and Main Result

We study a secure function computation problem involving three parties named Alice, Bob, and Charlie. Alice and Bob each have a sequence of n symbols, denoted respectively by $x^n := (x_1, \ldots, x_n) \in \mathcal{X}^n$ and $y^n := (y_1, \ldots, y_n) \in \mathcal{Y}^n$, where $\mathcal{X}$ and $\mathcal{Y}$ are finite alphabets. Charlie wishes to compute a function, $f_n(x^n, y^n)$, of Alice and Bob's sequences. The objective is to design a three-party protocol that allows Charlie to securely compute $f_n(x^n, y^n)$ with high accuracy and low communication cost. In the remainder of this section, we describe the class of functions of interest and make precise the notions of accuracy, security, and communication cost.

The class of functions that we consider are the *normalized sum-type functions*, $f_n : \mathcal{X}^n \times \mathcal{Y}^n \to \mathbb{Q}$, which are expandable in the form

$$f_n(x^n, y^n) = \frac{1}{n} \sum_{i=1}^{n} f_1(x_i, y_i),$$

for some function $f_1 : \mathcal{X} \times \mathcal{Y} \to \mathbb{Q}$.

A protocol is a sequence of instructions that the parties correctly follow. The execution of a protocol consists of a sequence of local computations and message transfers between the three parties via bi-directional, error-free channels that are available between each pair of parties. The messages sent at any stage of the execution of the protocol may depend on previously received messages, the sequences that are available to the parties sending the messages, and any independent local randomness that is generated. When the execution of the protocol terminates, an estimate $\hat{F}_n(x^n, y^n)$ is produced by Charlie. While the inputs (x^n, y^n) and the function f_n are deterministic, the estimate $\hat{F}_n(x^n, y^n)$ may be random due to inherent randomness in the protocol. We define the *view* of a party as the set of all messages sent or received, and any local randomness generated by that party during the execution of the protocol.

Accuracy: The distortion criterion to be minimized is the *maximal expected absolute error*, given by

$$e_n := \max_{x^n, y^n} E\left[\left|\hat{F}_n(x^n, y^n) - f_n(x^n, y^n)\right|\right],$$

where the expectation is with respect to the local randomness that is generated in the execution of the protocol. We emphasize that x^n and y^n are deterministic sequences and the distortion is the worst-case expected absolute error.

Security: We will consider protocols that achieve *unconditional privacy* for *semi-honest* parties. The semi-honest assumption means that the parties will correctly follow the protocol. Unconditional privacy against Alice and Bob means that after the execution of the protocol, the views of Alice and Bob do not reveal any information, in a strong statistical sense, about the other party's sequence. Unconditional privacy against Charlie means that after the execution of the protocol, the view of Charlie does not reveal any information, in a strong statistical sense, about (x^n, y^n) other what can be inferred from $\hat{F}_n(x^n, y^n)$. Let the random variables V_A, V_B, and V_C respectively denote the views of Alice, Bob, and Charlie after the execution of the protocol.

A protocol is *private against Alice* if the *distribution* of the view of Alice is only parameterized by x^n, that is, for all $(x^n, y^n, \bar{y}^n)$ in $\mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{Y}^n$,

$$P_{V_A}(v_a; x^n, y^n) = P_{V_A}(v_a; x^n, \bar{y}^n).$$

Similarly, a protocol is *private against Bob*, if the distribution of the view of Bob is only parameterized by y^n, that is, for all $(x^n, \bar{x}^n, y^n)$ in $\mathcal{X}^n \times \mathcal{X}^n \times \mathcal{Y}^n$,

$$P_{V_B}(v_b; x^n, y^n) = P_{V_B}(v_b; \bar{x}^n, y^n).$$

A protocol is *private against Charlie* if the conditional distribution of the view of Charlie given the estimate is not parameterized by x^n or y^n, that is, for all $(x^n, \bar{x}^n, y^n, \bar{y}^n)$ in $\mathcal{X}^n \times \mathcal{X}^n \times \mathcal{Y}^n \times \mathcal{Y}^n$,

$$P_{V_C|\hat{F}_n}(v_c|\hat{f}; x^n, y^n) = P_{V_C|\hat{F}_n}(v_c|\hat{f}; \bar{x}^n, \bar{y}^n).$$

A protocol is *unconditionally private* if it is private against Alice, Bob, and Charlie as defined above.

The security conditions are based on the strong notion of statistical indistinguishability. Note that for an unconditionally private protocol, if Alice and Bob's deterministic sequences (x^n, y^n) were replaced with random variables (X^n, Y^n) drawn from *any distribution*, then the views of each player would satisfy the following Markov chains,

$$V_A - X^n - Y^n,$$

from privacy against Alice,

$$V_B - Y^n - X^n,$$

from privacy against Bob, and

$$V_C - \hat{F}_n - (X^n, Y^n),$$

from privacy against Charlie. These Markov chains are analogous to the conditional mutual information conditions of [3] when appropriately adapted to our problem involving three semi-honest parties.

Communication Cost: For a given protocol, let k denote the number of bits necessary to send all of the messages required by the protocol. The communication cost of a protocol is given by the rate $R := (k/n)$.

Our main result is that any normalized sum-type function can be computed with arbitrarily high accuracy, vanishing communication cost, and unconditional privacy.

Theorem 2.1: There exist unconditionally private, randomized, three-party protocols such that, for all $m \in \{1, \ldots, n\}$, any normalized sum-type function can be computed with maximum expected absolute error bounded by

$$e_n \leq \frac{\|f_1\|_2}{\sqrt{m}},$$

and total communication cost on the order of

$$R = \frac{O(m \log n)}{n}.$$

By appropriately setting the parameter m we get the following corollary.

Corollary 2.1: By choosing a sequence of parameters m_n such that as n goes to ∞,

$$m_n \to \infty, \quad \frac{m_n \log n}{n} \to 0,$$

any normalized sum-type function can be computed with unconditional privacy, and maximal expected absolute error $e_n \to 0$ and communication cost $R \to 0$ as $n \to \infty$.

III. Proof of Theorem 2.1 on Vanishing Distortion and Vanishing Rate

We will prove the theorem by constructing protocols that attain the performance guarantees of Theorem 2.1. Our protocols produce a function estimate generated from only a random subsampling of the sequences (x^n, y^n). This technique utilizes the striking dimensionality reduction result of [7] that an accurate estimate of the joint type of (x^n, y^n) can be produced from only a random subsampling of the sequences (x^n, y^n). Additionally, we use the fact that a normalized sum-type function can be computed from the joint type. We will first discuss the dimensionality reduction result in Section III-A in order to analyze the function estimate in Section III-B. We will then construct unconditionally private protocols in Section III-C that securely produce this function estimate with vanishing communication cost on account of the random subsampling.

A. Estimating the Joint Type

Outside of the context of secure function computation, we first discuss a dimensionality reduction result concerning the estimation of the joint type (empirical distribution) $P_{x^n,y^n}(x, y)$ from only a random subsampling of the sequences (x^n, y^n). This subsampling technique and related results were first presented in [7]. Here we present an alternative and much simpler analysis, which allows us to compute bounds on the expected distortion and extend these results to the estimation of general normalized sum-type functions.

For a given subsampling parameter $m \in \{1, \ldots, n\}$, choose m locations from $\{1, \ldots, n\}$ uniformly *without* replacement. Let this index set of randomly chosen locations be denoted by the random variable I, and $(x_i, y_i)_{i \in I}$ denote the sequence subsampled from (x^n, y^n) according to the locations in I.

We define the full and partial frequency functions (histograms) $N, L : \mathcal{X} \times \mathcal{Y} \to \{0, \ldots, n\}$ according to

$$\begin{aligned} N(x, y) &:= nP_{x^n,y^n}(x, y), \\ L(x, y) &:= \big|\{i \in I : (x_i, y_i) = (x, y)\}\big|. \end{aligned}$$

Note that $N(x, y)$ is a deterministic quantity, while $L(x, y)$ is a hypergeometric random variable, since m samples are chosen without replacement from a set of n, where $N(x, y)$ of them can contribute to the value of $L(x, y)$.

Let the estimate of the joint type be given by

$$\hat{P}_{x^n,y^n}(x, y) := \frac{L(x, y)}{m}.$$

The mean and variance of this estimate are given by

$$\begin{aligned} E\big[\hat{P}_{x^n,y^n}(x,y)\big] &= \frac{E[L(x,y)]}{m} \\ &= \frac{N(x,y)}{n} \\ &= P_{x^n,y^n}(x,y), \\ \mathrm{Var}\big[\hat{P}_{x^n,y^n}(x,y)\big] &= \frac{\mathrm{Var}[L(x,y)]}{m^2} \\ &= \frac{N(x,y)(n-N(x,y))(n-m)}{mn^2(n-1)} \\ &\leq \frac{N(x,y)}{mn}. \end{aligned}$$

Thus, the mean squared error summed across (x,y) is given by

$$\begin{aligned} \Sigma_{\mathrm{MSE}} &:= E\left[\sum_{x,y}\left|\hat{P}_{x^n,y^n}(x,y) - P_{x^n,y^n}(x,y)\right|^2\right] \\ &= \sum_{x,y}\mathrm{Var}\big[\hat{P}_{x^n,y^n}(x,y)\big] \leq \frac{1}{m}. \end{aligned}$$

Continuing with Jensen's inequality yields a bound on the expected L_2 norm of the error,

$$E\left[\left\|\hat{P}_{x^n,y^n} - P_{x^n,y^n}\right\|_2\right] \leq \sqrt{\Sigma_{\mathrm{MSE}}} \leq \frac{1}{\sqrt{m}}.$$

Thus, for any pair of sequences (x^n,y^n) of any length n, an estimate of the joint type of the sequence-pair can be produced from only m random subsamples while achieving expected L_2-error inversely proportional to $\sqrt{m}$.

B. Obtaining the Function Estimate from the Joint Type

The random subsampling approach can be used to estimate any normalized sum-type function by expressing it as a function of the joint type as follows:

$$\begin{aligned} f_n(x^n,y^n) &= \frac{1}{n}\sum_{i=1}^{n} f_1(x_i,y_i) \\ &= \frac{1}{n}\sum_{x,y} f_1(x,y)N(x,y) \\ &= \sum_{x,y} f_1(x,y)P_{x^n,y^n}(x,y). \end{aligned}$$

Let an estimate of $f_n(x^n,y^n)$ based on only the subsampled sequence $(x_i,y_i)_{i\in I}$ be given by

$$\begin{aligned} \hat{F}_n(x^n,y^n) &:= \frac{1}{m}\sum_{i\in I} f_1(x_i,y_i) \qquad (1) \\ &= \frac{1}{m}\sum_{x,y} f_1(x,y)L(x,y) \qquad (2) \\ &= \sum_{x,y} f_1(x,y)\hat{P}_{x^n,y^n}(x,y). \end{aligned}$$

The absolute error of the function estimate

$$\begin{aligned} &\left|\hat{F}_n(x^n,y^n) - f_n(x^n,y^n)\right| \\ &\quad = \left|\sum_{x,y} f_1(x,y)\big(\hat{P}_{x^n,y^n}(x,y) - P_{x^n,y^n}(x,y)\big)\right| \\ &\quad \leq \left\|f_1\right\|_2 \cdot \left\|\hat{P}_{x^n,y^n} - P_{x^n,y^n}\right\|_2, \end{aligned}$$

by the Cauchy-Schwarz inequality, where

$$\left\|f_1\right\|_2 = \sqrt{\sum_{x,y}|f_1(x,y)|^2}.$$

Thus, the expected absolute error is bounded by

$$E\left[\left|\hat{F}_n(x^n,y^n) - f_n(x^n,y^n)\right|\right] \leq \frac{\|f_1\|_2}{\sqrt{m}}.$$

C. Function Evaluation Protocols

We provide three protocols, all of which make use of unconditionally secure multiparty computation methods to produce $\hat{F}_n(x^n,y^n)$ as given by (1). The common first step is for Alice to randomly choose the m subsampling locations $I \subset \{1,\ldots,n\}$, uniformly without replacement, and communicate them to Bob with $m\log n$ bits. From here, the specifics of the protocols differ, but they all require Alice and Bob to work with only the subsampled sequences $(x_i,y_i)_{i\in I}$ and result in Charlie computing $\sum_{i\in I} f_1(x_i,y_i) = m\hat{F}_n(x^n,y^n)$ via finite field arithmetic. Since the domain $\mathcal{X}\times\mathcal{Y}$ is finite, the range of f_1 is a finite subset of $\mathbb{Q}$. Thus, with a sufficiently large finite field, $\mathcal{F}_m$, to prevent interger-arithmetic overflow, the computation of $\sum_{i\in I} f_1(x_i,y_i)$ can be performed with finite field arithmetic in $\mathcal{F}_m$. The finite field representation of $\sum_{i\in I} f_1(x_i,y_i)$ can then be converted back into a rational number and divided by m to produce $\hat{F}_n(x^n,y^n)$.

All three protocols require $O(m\log|\mathcal{F}_m|)$ bits in addition to the $m\log n$ bits required to transmit I. Since the size of the finite field need only be on the order[2] of $|\mathcal{F}_m| = O(m)$, the total number of bits needed is actually dominated by the transmission of I and is on the order of $k = O(m\log n)$. We will discuss and compare the specific communication cost of each protocol. All of the protocols are unconditionally private, however detailed proofs of privacy are omitted due to length restrictions.

D. One-Time Pad Protocol

Our first protocol leverages a type of homomorphism achievable with one-time pad encryption. Alice and Bob respectively send their subsampled sequences $(x_i)_{i\in I}$ and $(y_i)_{i\in I}$, masked (encrypted) with one-time pads, to Charlie. From these encrypted sequences, Charlie computes and returns to Alice and Bob encrypted additive shares of the partial frequency function L. After exchanging their one-time pads, Alice and Bob decrypt their respective messages from Charlie to obtain the additive shares of L, from which

[2]The computation can be expressed in a finite field of a size on the order of $O(m)$ since it is a sum of m rational values from the finite image set of f_1.

TABLE I

COMPARISON OF THREE-PARTY PROTOCOLS FOR SECURELY COMPUTING NORMALIZED SUM-TYPE FUNCTIONS

Protocol	Bits required in addition to $m \log n$	Advantages
1) One-Time Pad	$2m(\log \lvert\mathcal{X}\rvert + \log \lvert\mathcal{Y}\rvert + \lvert\mathcal{X}\rvert\lvert\mathcal{Y}\rvert \log \lvert\mathcal{F}_m\rvert) + 3 \log \lvert\mathcal{F}_m\rvert$	Simplest techniques: one-time pads, additive shares
2) Poly Secret-Share L	$(2m(\lvert\mathcal{X}\rvert + \lvert\mathcal{Y}\rvert) + 2) \log \lvert\mathcal{F}_m\rvert$	Most efficient for complex f_1
3) Poly Secret-Share Direct	(Varies), at best $(4m + 2) \log \lvert\mathcal{F}_m\rvert$	Most efficient for simple f_1

they derive additive shares of the function estimate that are returned to be recombined by Charlie. This technique of first computing additive shares of L, as an intermediate step, takes advantage of the function estimate expansion given by (2). The transmissions required by and the complexity of implementing this protocol are independent of the complexity of f_1 (except indirectly through the necessary size of $\mathcal{F}_m$).

The detailed steps of this protocol are:

1) Alice generates a one-time pad $(\alpha_i)_{i \in I}$, by choosing $\alpha_i \sim$ iid Unif$(\{0, \ldots, |\mathcal{X}| - 1\})$. The pad is applied to $(x_i)_{i \in I}$, producing the encrypted sequence $(\overline{x}_i)_{i \in I}$, by setting $\overline{x}_i = \oplus_{\alpha_i}(x_i)$, where $\oplus_{\alpha_i}(x_i)$ is a circular shift of the value of x_i over an arbitrary ordering of $\mathcal{X}$ by α_i positions.
2) Similarly, Bob generates a one-time pad $(\beta_i)_{i \in I}$, with $\beta_i \sim$ iid Unif$(\{0, \ldots, |\mathcal{Y}| - 1\})$, and applies it to $(y_i)_{i \in I}$ to produce $(\overline{y}_i)_{i \in I}$ by setting $\overline{y}_i = \oplus_{\beta_i}(y_i)$.
3) Alice and Bob respectively send $(\overline{x}_i)_{i \in I}$ and $(\overline{y}_i)_{i \in I}$ to Charlie, using a total of $m(\log |\mathcal{X}| + \log |\mathcal{Y}|)$ bits.
4) For each $i \in I$, Charlie produces M_i, an $|\mathcal{X}| \times |\mathcal{Y}|$ matrix indexed by $(x, y) \in \mathcal{X} \times \mathcal{Y}$, where $M_i(x, y) = \mathbf{1}_{\{\overline{x}_i, \overline{y}_i\}}(x, y)$, which is the indicator function equal to one if $(\overline{x}_i, \overline{y}_i) = (x, y)$ and zero otherwise.
5) Charlie splits each M_i into additive shares, by first independently choosing, across all $(i, x, y) \in I \times \mathcal{X} \times \mathcal{Y}$, $M_{A,i}(x, y) \sim$ iid Unif$(\mathcal{F}_m)$, then computing $M_{B,i} = M_i - M_{A,i}$.
6) Charlie sends the matrices $(M_{A,i})_{i \in I}$ to Alice and $(M_{B,i})_{i \in I}$ to Bob, using $2m|\mathcal{X}||\mathcal{Y}| \log |\mathcal{F}_m|$ bits (in total).
7) Alice and Bob exchange their one-time pads, $(\alpha_i)_{i \in I}$ and $(\beta_i)_{i \in I}$, using $m(\log |\mathcal{X}| + \log |\mathcal{Y}|)$ bits.
8) Alice and Bob separately decrypt $(M_{A,i})_{i \in I}$ and $(M_{B,i})_{i \in I}$ to compute additive shares of L, via

$$L_A(x, y) = \sum_{i \in I} M_{A,i}(\oplus_{\alpha_i}(x), \oplus_{\beta_i}(y)),$$
$$L_B(x, y) = \sum_{i \in I} M_{B,i}(\oplus_{\alpha_i}(x), \oplus_{\beta_i}(y)).$$

9) Alice and Bob separately compute additive shares of the function computation, via

$$F_A = \sum_{x,y} f_1(x, y) L_A(x, y),$$
$$F_B = \sum_{x,y} f_1(x, y) L_B(x, y).$$

10) Alice independently generates random salt[3] Z uniformly over $\mathcal{F}_m$, which she sends to Bob using $\log |\mathcal{F}_m|$ bits.
11) Alice and Bob send $F_A + Z$ and $F_B - Z$ to Charlie using a total of $2 \log |\mathcal{F}_m|$ bits. Note that $F_A + F_B = m\hat{F}_n(x^n, y^n)$ because of the definition of $M_i(x, y)$. Thus, Charlie can produce $\hat{F}_n(x^n, y^n)$.

Thus, in addition to the $m \log n$ bits needed to convey the sampling set I from Alice to Bob, this protocol requires an additional $2m(\log |\mathcal{X}| + \log |\mathcal{Y}| + |\mathcal{X}||\mathcal{Y}| \log |\mathcal{F}_m|) + 3 \log |\mathcal{F}_m|$ bits, which is on the order of $O(m \log |\mathcal{F}_m|)$.

E. Polynomial Secret-Sharing Protocols

Our next two protocols employ the secure multi-party computation methods of [1], which exploit the homomorphic properties of polynomial-based secret sharing [11].

Our second protocol takes an approach quite similar to the first but replaces the homomorphism achieved with one-time pad encryption with the secure multi-party computation methods of [1]. The three parties first compute homomorphic shares of the partial frequency function L, from which homomorphic shares of the function computation can be obtained.

The detailed steps of this protocol are:

1) For each $(i, x) \in I \times \mathcal{X}$, Alice independently chooses $\alpha_{ix} \sim$ iid Unif$(\mathcal{F}_m)$ and constructs the polynomial

$$g_{ix}(p) = \alpha_{ix} p + \mathbf{1}_{\{x_i\}}(x),$$

where $\mathbf{1}_{\{x_i\}}(x)$ is the indicator function equal to 1 if $x_i = x$ and 0 otherwise. For each $(i, x) \in I \times \mathcal{X}$, Alice sends the sample $g_{ix}(2)$ to Bob and the sample $g_{ix}(3)$ to Charlie using $2m|\mathcal{X}| \log |\mathcal{F}_m|$ bits, while keeping the sample $g_{ix}(1)$ for herself.
2) For each $(i, y) \in I \times \mathcal{Y}$, Bob independently chooses $\beta_{iy} \sim$ iid Unif$(\mathcal{F}_m)$ and constructs the polynomial

$$h_{iy}(p) = \beta_{iy} p + \mathbf{1}_{\{y_i\}}(y),$$

where $\mathbf{1}_{\{y_i\}}(y)$ is the indicator function equal to 1 if $y_i = y$ and 0 otherwise. For each $(i, y) \in I \times \mathcal{Y}$, Bob sends the sample $h_{iy}(1)$ to Alice and the sample $h_{iy}(3)$ to Charlie using $2m|\mathcal{Y}| \log |\mathcal{F}_m|$ bits, while keeping the sample $h_{iy}(2)$ for himself.
3) Each party can compute a sample of the polynomial given by

$$F(p) = \sum_{x,y} f_1(x, y) \sum_{i \in I} g_{ix}(p) h_{iy}(p).$$

[3]The random salt is used to maintain security by statistically decorrelating Alice and Bob's function estimate shares from information already held by Charlie.

Alice can compute $F(1)$ from her samples of $g_{ix}(1)$ and $h_{iy}(1)$. Likewise, Bob can compute $F(2)$ and Charlie can compute $F(3)$.

4) It follows that

$$\begin{aligned} F(0) &= \sum_{x,y} f_1(x,y) \sum_{i \in I} \mathbf{1}_{\{x_i,y_i\}}(x,y) \\ &= m\hat{F}_n(x^n, y^n), \end{aligned}$$

and that $F(p)$ is a degree-two polynomial of the variable p. The values of $F(1)$ and $F(2)$ are sent, using $2\log|\mathcal{F}_m|$ bits, to Charlie, who already has $F(3)$. Via polynomial interpolation, Charlie produces $F(0) = m\hat{F}_n(x^n, y^n)$ and hence $\hat{F}_n(x^n, y^n)$.

Thus, in addition to the $m \log n$ bits needed to convey the sampling set I from Alice to Bob, this protocol requires an additional $(2m(|\mathcal{X}|+|\mathcal{Y}|)+2)\log|\mathcal{F}_m|$ bits, which is on the order of $O(m \log|\mathcal{F}_m|)$.

Our final protocol takes a direct approach toward computing $\sum_{i\in I} f_1(x_i, y_i)$, by first using the secure computation methods of [1] to compute homomorphic shares of $f_1(x_i, y_i)$, which can then be summed across $i \in I$ in order to produce shares of the function estimate. This approach directly reflects the expansion of the function estimate given by (1).

The main steps of this protocol are as follows:

1) For each $i \in I$, Alice, Bob, and Charlie use the secure computation methods of [1] to respectively obtain shares $f_A(x_i, y_i)$, $f_B(x_i, y_i)$, and $f_C(x_i, y_i)$, which are samples of a random polynomial with a zero-order coefficient equal to $f_1(x_i, y_i)$.
2) Alice, Bob, and Charlie respectively compute shares F_A, F_B, and F_C via

$$\begin{aligned} F_A &= \sum_{i\in I} f_A(x_i, y_i), \\ F_B &= \sum_{i\in I} f_B(x_i, y_i), \\ F_C &= \sum_{i\in I} f_C(x_i, y_i), \end{aligned}$$

which are now samples of a random polynomial with a zero-order coefficient equal to $m\hat{F}_n(x^n, y^n)$.
3) The values F_A and F_B are sent, using $2\log|\mathcal{F}_m|$ bits, to Charlie who interpolates the polynomial in order to produce $m\hat{F}_n(x^n, y^n)$, and hence $\hat{F}_n(x^n, y^n)$.

Note that the complexity of this protocol is captured in the first step of computing shares of $f_1(x_i, y_i)$. The actual details of this step depends on the structure of f_1 and how it can be represented by a multi-variate polynomial over a finite field, which, in principle, is always feasible by interpolation but could possibly result in a very complex polynomial. The first step will require at least $4m \log|\mathcal{F}_m|$ bits of transmission for Alice and Bob to initially distribute shares of $(x_i, y_i)_{i\in I}$. However, additional transmissions could be necessary in order to perform the degree reduction and randomization steps needed after each multiplication in the polynomial expressing f_1 (see [1] for details). While the actual transmission cost would depend on the complexity of the polynomial realizing f_1, the cost would be proportional to $m \log|\mathcal{F}_m|$ times the multiplicative depth of the polynomial representation of f_1. This last protocol requires at best, an additional $(4m+2)\log|\mathcal{F}_m|$ bits, however even at worst, the additional bits required is still on the order $O(m \log|\mathcal{F}_m|)$.

F. Comparison of Protocols

All of our protocols are unconditionally private and produce the same estimate $\hat{F}_n(x^n, y^n)$ while requiring $m \log n + O(m \log|\mathcal{F}_m|)$ bits. Their subtle performance differences are in the constants of the $O(m \log|\mathcal{F}_m|)$ term. For functions where f_1 can be represented as a polynomial with a multiplicative depth of one, the third protocol is the simplest and most efficient, using only $(4m+2)\log|\mathcal{F}_m|$ bits in addition to the $m \log n$ needed to trasnmit I. However, for functions where f_1 is more complicated (e.g., containing absolute values or thresholding), and needs a polynomial representation of multiplicative depth greater than one, the complexity and the bits needed for the third protocol increase. For such functions it is better to use the first and second protocols which compute, as an intermediate step, homomorphic shares of the partial frequency function L. The complexity and bits required by the first two protocols are not affected by the complexity of f_1 (except indirectly through the necessary size of $\mathcal{F}_m$), and hence are more efficient than the third protocol for very complex functions f_1. The first protocol is less efficient than the second, but is of interest since it demonstrates how the simple techniques of one-time pad encryption and additive shares are sufficient to construct a secure computation protocol for this problem. Table I summarizes the advantages and communication costs of the three protocols.

IV. Final Remarks

We have introduced a distortion-theoretic approach for secure multi-party computation with unconditional privacy. By extending the dimensionality reduction result of [7], we have constructed protocols that securely compute any normalized sum-type function with arbitrarily high accuracy and vanishing communication cost. The technique of randomized subsampling allowed us to overcome the worst-case distortion criterion, yielding the result that for any sequences (x^n, y^n) of any arbitrary length n, the expected absolute error of the function estimate constructed from only m random subsamples is inversely proportional to $\sqrt{m}$.

For clarity of exposition, we only considered the scenario in which Charlie wished to compute a function of Alice and Bob's sequences. However, the protocols are easily modified to allow each party to securely compute a unique normalized sum-type function. The security conditions are also easily modified to reflect this scenario. The results can also be extended to more than three parties, by constructing protocols using the methods of [1] along with randomized subsampling to provide dimensionality reduction. To extend these results to a two-party scenario, the randomized subsampling technique could be paired with secure function computation techniques that utilize an oblivious transfer primitive [12] or

a binary erasure channel [4]. A notion of communication cost similar to [4] could be defined by counting the number of erasure channel uses or oblivious transfer primitive uses and dividing by n. In these extensions, it would also be possible to prove similar results on achieving vanishing distortion and vanishing communication cost, while maintaining unconditional security.

REFERENCES

[1] M. Ben-Or, S. Goldwasser, A. Wigderson, "Completeness theorems for Non-Cryptographic Fault-Tolerate Distributed Computation," in *Proc. ACM STOC*, 1988, pp. 1–10.

[2] D. Chaum, C. Crépeau, I. Damgård, "Multi-Party Unconditionally Secure Protocols," in *Proc. ACM STOC*, 1988, pp. 11–19.

[3] C. Crépeau, G. Savvides, C. Schaffner, and J. Wullschleger, "Information-theoretic conditions for two-party secure function evaluation," in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 538–554.

[4] Y. Wang and P. Ishwar, "Bootstrapped Oblivious Transfer and Secure Two-Party Function Computation," in *Proc. IEEE International Symposium on Information Theory*, Seoul, South Korea, Jun. 2009.

[5] P. Indyk and J. Matousek, "Low-distortion embeddings of finite metric spaces," in *Handbook of Discrete and Computational Geometry*. CRC Press, 2004, pp. 177–196.

[6] W. B. Johnson and J. Lindenstrauss, "Extensions of Lipshitz mapping into Hilbert space," *Contemporary Mathematics*, vol. 26, pp. 189–206, 1984.

[7] R. Ahlswede and Z. Zhang, "Estimating with Randomized Encoding the Joint Empirical Distribution in a Correlated Source," in *Lecture Notes in Computer Science*, vol. 4123. Berlin, Germany: Springer, 2006, pp. 535–546.

[8] R. Ahlswede and I. Csiszár, "To Get a Bit of Information May Be As Hard As to Get Full Information," *IEEE Trans. Info. Theory*, vol. IT–27, no. 4, pp. 398–408, Jul. 1981.

[9] A. El Gamal, "A Simple Proof of the Ahlswede–Csiszár One-Bit Theorem," *IEEE Trans. Info. Theory*, vol. IT–29, no. 6, pp. 931–933, Nov. 1983.

[10] S. Rane, W. Sun, and A. Vetro, "Secure Distortion Computation in the Encrypted Domain Using Homomorphic Encryption," in *Proc. IEEE International Conference on Image Processing*, Cairo, Egypt, Oct. 2009.

[11] A. Shamir, "How to Share a Secret," *Communications of the ACM*, vol. 22, no. 11, pp. 637–647, 1985.

[12] J. Kilian, "Founding Cryptography on Oblivious Transfer," in *Proc. 20th ACM Symp. Theory of Comp. (STOC)*, Chicago, IL, 1988, pp. 20–31.